CENTER FOR PURE AND APPLIED MATHEMATICS
UNIVERSITY OF CALIFORNIA, BERKELEY

PAM-587

IMPLICIT CHOLESKY ALGORITHMS FOR SINGULAR VALUES AND VECTORS

K. Vince Fernando and Beresford N. Parlett

August 1993

This report was done with support from the Center for Pure and Applied Mathematics. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of the Center for Pure and Applied Mathematics or the Department of Mathematics.

DEPARTMENT OF THE NAVY
OFFICE OF NAVAL RESEARCH
SEATTLE REGIONAL OFFICE
1107 NE 45TH STREET, SUITE 350
SEATTLE WA 98105-4631

IN REPLY REFER TO:

4330
ONR 247
11 Jul 97

From: Director, Office of Naval Research, Seattle Regional Office, 1107 NE 45th St., Suite 350, Seattle, WA 98105
To: Defense Technical Center, Attn: P. Mawby, 8725 John J. Kingman Rd., Suite 0944, Ft. Belvoir, VA 22060-6218

Subj: RETURNED GRANTEE/CONTRACTOR TECHNICAL REPORTS

1. This confirms our conversations of 27 Feb 97 and 11 Jul 97. Enclosed are a number of technical reports which were returned to our agency for lack of clear distribution availability statement. This confirms that all reports are unclassified and are "APPROVED FOR PUBLIC RELEASE" with no restrictions.

2. Please contact me if you require additional information. My e-mail is *silverr@onr.navy.mil* and my phone is (206) 625-3196.

[illegible]
ROBERT J. SILVERMAN

Implicit Cholesky Algorithms for Singular Values and Vectors

K Vince Fernando[1,2,a,b] and Beresford N Parlett[3,b]

[1] NAG Ltd, Jordan Hill, Oxford OX2 8DR, UK

[2] Division of Computer Science, University of California, Berkeley, CA 94708, USA

[3] Department of Mathematics, University of California, Berkeley, CA 94720, USA

July 29, 1993

[a]Supported by NSF, under grant ASC-9005933
[b]Supported by ONR, contract N000014-90-J-1372

Abstract

The implicit Cholesky algorithm has been developed by several authors during the last 10 years but under different names. We identify the algorithm with a special version of Rutishauser's LR algorithm. Intermediate quantities in the transformation furnish several attractive approximations to the smallest singular value.

The paper extols the advantages of using shifts with the algorithm. The non-orthogonal transformations improve accuracy.

Key words: Cholesky decomposition, singular values, Singular value decomposition, eigenvalues, null spaces, noise spaces, URV factorization, ULV factorization, QR algorithm, LR algorithm, Jacobi methods

Contents

1 Introduction

We want to recommend a new way to compute singular values and vectors of triangular matrices.

One of the ideas behind our algorithm is not new. The flipping of a triangular matrix between upper and lower forms, using orthogonal transformations exclusively, first from one side and then from the other has become a popular activity among signal processors recently, see [13], [4]. The end product is either the SVD or the updating of a noise subspace. Mathias and Stewart in [13], used this see-saw procedure in the refinement step of the URV factorization. Chandrasekaran and Ipsen in [2] looked at this flip-flop technique as the basic QR algorithm and Ammann in [1] goes so far as to call it the transpose QR algorithm. This name is a bit misleading.

As long ago as 1968 Faddeev et al, in [5] recognized that one such transformation of a triangular matrix is *equivalent* to one step of the Cholesky LR algorithm of Rutishauser: $L = QR$ implies $L^tL = R^tR$. However the way R is produced from L is quite different from Rutishauser's transformation of LL^t to R^tR and this suggests to us that the proper name for the triangular flipping procedure is the implicit Cholesky LR algorithm or, better, the **implicit Cholesky algorithm** since LR is a bit redundant. We plan to discuss the implicit LR algorithm for nonsymmetric eigenvalue problems in another communication. Our naming conventions are consistent with past usage; QR factorization is the basis for the QR algorithm. Similarly Cholesky factorization leads to the Cholesky algorithm. The interesting dissertation [22] adopts this terminology.

Since two steps of the unshifted Cholesky algorithm are equivalent to one step of the unshifted QR algorithm it is clear how QR comes into the picture.

What seems to be a novel contribution of this paper is to show how to combine the use of shifts and the calculation of either right or left singular vectors but not both. In many applications this suffices. Our Algorithm 5 only delivers the right singular vectors.

We discuss convergence to show how easy the argument is in the unshifted case and to sharpen the idea of Rutishauser for getting a very accurate shift from a case of late failure in the shifted version.

We do not give a formal error analysis here because the backward stability results for the unshifted case extend to our shifted algorithm. In fact our accuracy (absolute, not relative) is better than the standard procedures in practice but we have not proved this property yet. Our shifted algorithms do not preserve the Frobenius norm they actually decrease it. This improves speed and accuracy.

There is some evidence that the connection between triangular flipping and either QR or Cholesky is not widely appreciated. The strongest piece of evidence is the absence of shifting. Experts are happily extolling the flip-flop of triangular matrices who would not be caught dead using QR or LR algorithms without appropriate shifts. It is as though an interdiction has been placed on using anything other than orthogonal transformations for SVD calculations. In [7] we showed that for high relative accuracy in treating bidiagonal matrices it is advisable to abandon plane rotations. Consequently orthogonal transformations are sufficient but by no means necessary for obtaining accurate approximations.

The numerical results suggest to us that the current triangular flipping algorithms are popular because little accuracy is required and the snail like pace of linear convergence does not hurt much. With sensible shifts the implicit Cholesky algorithm can be used like a LAPACK routine to yield singular values with maximal accuracy in an absolute sense ($macheps * norm$).

Here we present the implicit Cholesky algorithm with shifts and make a plea for its adoption, at least by those who are in a hurry. New results, Theorems 1 and 3, show that the intermediate quantities produced by the algorithm yield valuable information for shift selection.

2 Notation

Householder conventions are followed: capital letters for matrices, Greek letters for scalars, lower case Roman for vectors. We use the Euclidean vector norm $\|x\|^2 = x^*x$ and the spectral matrix norm $\|F\| = \sigma_1[F]$ throughout the paper where $\sigma_i[.]$ denotes the ith singular value. Similarly, $\lambda_i[.]$ signifies the ith eigenvalue. We use x^* for the conjugate transpose of x but since most of our quantities are real x^* also denotes the transpose.

The columns of the identity matrix are denoted by e: $I = (e_1, e_2, \ldots, e_n)$.

3 Uses of Triangular Form

If one wants to compute the SVD of a single matrix, the most efficient and accurate method available appears to be the shifted differential qd algorithm of Fernando and Parlett which is a special case of the implicit LR algorithm. See [7]. However this requires the bidiagonalization of the matrix using orthogonal transformations. There are many occasions when this bidiagonalization is not warranted. For example, in signal processing applications this reduction is not advised since new data has to be brought in and old data has to be removed. Triangular matrices are a convenient form in this case. See for example [14] and [20].

Although the SVD is the most reliable decomposition to obtain the rank of a matrix, it is often an overkill. That is why intermediate non-canonical decompositions such as the URV and ULV factorizations of Stewart have many applications, especially in signal processing and statistics. If one requires only the smallest singular values or the singular vectors corresponding to the smallest singular values, the triangular structure is often adequate to obtain these rank revealing properties. In addition the condition number is revealed very quickly with our shifted implicit Cholesky algorithm.

The reduction to bidiagonal form may not be needed if the matrix already has small off-diagonal elements and the diagonal entries are monotone decreasing or nearly so.

The flipping process can be also used as a preconditioner for other SVD algorithms. Veselić and Hari used one step of the Cholesky algorithm with pivoting as an effective preconditioner for the one-sided Jacobi algorithm of Hestenes. Lacking the implicit form of the algorithm they felt obliged to limit themselves to one preconditioning step. However, by using the flipping process many preconditioning steps can be used.

If the matrix is banded and triangular, then the flipping process preserves bandwidth. This comes from the well known fact that the LR algorithm preserves the bandwidths of matrices. Since bidiagonalization of banded matrices is relatively expensive, implicit Cholesky is a convenient method to find the SVD or the approximate null space of a banded matrix.

4 The Implicit Cholesky Algorithm

Our goal here is to present neglected implementations of the Cholesky LR algorithm that avoid the two defining steps of this classical method presented as Algorithm 1.

Algorithm 1 (Standard Cholesky to diagonalize symmetric positive definite A)

```
$A_0 := A$
For $i = 0, 1, 2, \dots$ until converged
    (a) Compute the Cholesky factorization $L_i L_i^* := A_i$
    (b) $A_{i+1} \leftarrow L_i^* L_i$
```

Here L_i is lower triangular and L_i^* is its conjugate transpose.

In exact arithmetic as $i \to \infty$, $L_i \to \Sigma$, the diagonal matrix of (often, ordered) singular values of L_0. Moreover, $A_i \to \Sigma^2$, the diagonal matrix of ordered eigenvalues of A. In floating-point arithmetic, the computed eigenvalues will be accurate in an absolute sense but the relative accuracy of smaller singular values may be poor.

The overlooked fact is that there is no need to form A_{i+1} in order to obtain its Cholesky factor L_{i+1}. Instead consider the QR factorization of L_i, say

$$L_i = Q_i R_i$$

where $Q_i^* = Q_i^{-1}$ and R_i is upper triangular with positive diagonal.

Lemma 1

$$L_{i+1} = R_i^*$$

Proof: By step (b) of the algorithm

$$A_{i+1} = L_i^* L_i = R_i^* Q_i^* Q_i R_i = R_i^* R_i$$

By step (a)

$$A_{i+1} = L_{i+1} L_{i+1}^*$$

and the result follows from the uniqueness of the Cholesky factorization. •

There is considerable computational advantage in not forming the matrices A_i.

Comment It seems likely that Rutishauser would have discovered Lemma 1 (via the qd algorithm, see [16], [18], [17], [19]) but we can find no explicit mention of this fact.

A defect of the traditional LR algorithm is that one cannot obtain either eigenvectors of A or the singular vectors of L_0 directly. However the new implementation provides a natural mechanism for obtaining those vectors, as shown below. To see the idea write out the relation of L_2 to L_0,

$$L_2 = L_1^* Q_1 = Q_0^* L_0 Q_1.$$

Similarly,

$$L_4 = L_3^* Q_3 = \ldots = Q_2^* Q_0^* L_0 Q_1 Q_3.$$

We now describe the new implementation formally.

Algorithm 2 (Implicit Cholesky SVD of L)

```
Set $L_0 = L$, $U = I$, $V = I$
For $i = 1, 2, \ldots$ until converged
    (a) Compute the QR factorization $L_i = Q_i R_i$,
    (b) $U \leftarrow U Q_i$ for odd $i$, $V \leftarrow V Q_i$ for even $i$
    (c) $L_{i+1} \leftarrow R_i^*$. (This step is purely formal)
```

Remark 1 It is not necessary to form Q_i explicitly to obtain R_i $(= L_{i+1}^*)$.

Remark 2 On exit the squares of L's diagonals give the eigenvalues of $L_0 L_0^* (= A_0)$.

Remark 3 Also, on exit, U holds the left singular vectors of L, the normalized eigenvectors of $L_0 L_0^*$. If U is not required, that part of the algorithm may be omitted.

Remark 4 If the right singular vector matrix V is not wanted, that part of the algorithm may be omitted.

Remark 5 If the matrix A_0 is positive semi-definite , then the traditional LR algorithm could break down. However, the **Implicit Cholesky** algorithm exists even when L_0 is not full rank. Diagonal pivoting cures the defect. See Section 5.

Remark 6 The identification of the basic algorithm with unshifted QR (see next section) indicates that the process is rather slow. Nevertheless if L nearly reveals its rank then only a few steps are needed to obtain a basis for the approximate null space.

Remark 7 Since Cholesky factorization preserves band structure and so does reverse multiplication of the factors it is clear that Algorithm 1 preserves bandwidth. Consequently Algorithms 2 and 3 enjoy the same property.

5 The Relation of QR to Implicit Cholesky

It is well known - to experts - that, for a positive definite Hermitian matrix, two steps of the LR algorithm produce the same matrix as one step of the zero-shift QR algorithm. For completeness we show this well known relationship which is hard to find in the literature.

Algorithm 3 (Zero-shift QR algorithm for Hermitian M)

```
$M_0 = M$.
For $i = 0, 1, 2, \dots$ until converged
    (a) Compute the QR factorization $M_i = Z_i T_i$,
    (b) $M_{i+1} \leftarrow T_i Z_i$ .
```

Theorem 1 *One step of the QR algorithm with zero shifts is equivalent to two steps of the unshifted* ***Implicit Cholesky****.*

Proof: Set $M_0 = A_0 = L_0 L_0^*$. By Algorithm 2,

$$M_0 = L_0 L_0^* = Q_0 L_1 L_0^*.$$

By uniqueness of the QR factorization $Z_0 = Q_0$ and $T_0 = L_1^* L_0^*$ and so, from proof of Lemma 1,

$$L_2 L_2^* = Q_0^* L_0 L_0^* Q_0 = Z_0^* M_0 Z_0 = M_1.$$

Similarly,

$$A_{2j+2} = L_{2j+2} L_{2j+2}^* = Q_{2j}^* L_{2j} L_{2j}^* Q_{2j} = Z_j^* M_j Z_j = M_{j+1} \bullet$$

6 Incorporation of Pivoting

It is recognized that pivoting is useful for improving convergence of the basic LR algorithm [18], [17]. In the case of the classical LR algorithm pivoting is used to bring the large diagonal elements to the top of the matrix and the small diagonal values to the bottom of the matrix.

For Algorithm 2, we can accomplish this by using the pivoted QR factorization with column interchanges,

$$L_i = Q_i R_i P_i$$

where P_i is a permutation matrix. Any rank revealing P_i could be used.

The relations are then

$$L_i P_i^* = Q_i L_{i+1}^*$$

We note that pivoting is difficult in many modern machines and in some novel architectures this could be prohibitively expensive. However, we do not expect pivoting to be needed except in the first or the second sweep of the algorithm for most matrix problems.

7 Bounds for σ_{min}

Useful bounds may be attained at little cost by exploiting intermediate quantities that occur in one step of the unshifted implicit Cholesky transform, $L = Q\hat{L}^*$.

Consider an intermediate stage in the transformation of L to $\hat{L}$ where $Q^*L = \hat{L}^*$. Although Q is unique it may be obtained as a product in many different ways. However the choice narrows down when it is desirable to take advantage of the triangular nature of L.

The first stage of the reduction L to $\hat{L}^*$ is typical. Apply a sequence of $(n-1)$ plane rotations in the planes $(1,j)$, $j = 2,\ldots,n$ to map column 1 of L into a multiple of $e_1 = (1,0,\ldots,0)^*$. Call the product of these rotations G_1^*, then

$$G_1^* L = \begin{bmatrix} \|l_1\| & e_1^*\hat{L}^* \\ 0 & L^{(2)} \end{bmatrix}$$

When the plane rotations are taken in the natural order indicated above then the "reduced matrix" $L^{(2)}$ must be lower triangular and may be transformed the same way. Note that use of a Householder reflector instead of G_1 would destroy the triangular form of $L^{(2)}$.

<u>Definition</u> The $(1,1)$ entry of the reduced matrix $L^{(k)}$ is denoted by d_k. Note that $d_1 = l_{1,1}$. •

Consider an intermediate stage in the reduction.

$$G_{k-1}^* \ldots G_1^* L = \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & d_k & & \\ & & & \times & \times & \\ & & & \times & \times & \times \end{pmatrix} \qquad (1)$$

Note that row k of the matrix is a singleton.

Theorem 1 *Transform $n \times n$ invertible lower triangular L to upper triangular $\hat{L}^*$ by the triangle preserving algorithm indicated in equation (1). Then*

(a) $\sigma_n[L] \le \min_k d_k$,

(b) $[(L^*L)^{-1}]_{k,k} = d_k^{-2}$,

(c) $(\sum_{k=1}^n d_k^{-1})^{-1} \le (\sum_{k=1}^n d_k^{-2})^{-\frac{1}{2}} \le \sigma_n[L]$.

Proof: The key fact is that, for each k,

$$e_k^* G_{k-1}^* \dots G_1^* L = d_k e_k^*. \qquad (2)$$

Since singular values are invariant under unitary equivalences

$$\sigma_n[L] = \sigma_n[G_{k-1}^* \dots G_1^* L] \le \|e_k^* G_{k-1}^* \dots G_1^* L\| = d_k,$$

yielding (a). Next transpose (2) and rearrange:

$$G_1 \dots G_{k-1} e_k d_k^{-1} = L^{-*} e_k,$$

$$d_k^{-2} = d_k^{-1} e_k^* G_{k-1}^* \dots G_1^* G_1 \dots G_{k-1} e_k d_k^{-1} = e_k^* L^{-1} L^{-*} e_k,$$

yielding (b). Finally summing the last equation over k gives

$$\sigma_n^{-2} \le \sum_{i=1}^n \sigma_i^{-2} = \|L^{-1}\|_F^2 = \text{trace}[(L^*L)^{-1}] = \sum_{k=1}^n d_k^{-2},$$

which is the last inequality in (3). The first inequality holds for any set of positive numbers: $\|v\|_2^2 \le \|v\|_1^2$. •

Remark By Theorem 2 we may use $\tau = (\sum_{k=1}^n d_k^{-2})^{-\frac{1}{2}}$ as a shift with no fear of breakdown. τ^2 is the Newton shift from 0 towards $\sigma_{min}^2[L]$. The reason is well known.

Theorem 2 *Let A be symmetric positive definite. The Newton approximation to $\lambda_{min}[A]$ from 0 is $1/trace(A^{-1})$.*

Proof: If $\chi(t)$ is A's characteristic polynomial then

$$-\chi'(0)/\chi(0) = \sum_{i=1}^n \frac{1}{\lambda_i - 0} = \text{trace}(A^{-1}) ,$$

where $\chi'(t)$ denotes the derivative of $\chi(t)$ with respect to t. •

8 Incorporating Shifts

Given L and $\tau < \sigma_{min}[L]$ one seeks $\hat{L}$, lower triangular with positive diagonal, such that

$$L^*L - \tau^2 I = \hat{L}\hat{L}^*.$$

Take the shift to the other side and observe that it is equivalent to compute an orthogonal $2n \times 2n$ matrix Q such that

$$Q\begin{bmatrix} L \\ 0 \end{bmatrix} = \begin{bmatrix} \hat{L}^* \\ \tau I \end{bmatrix}$$

Let m_{*j} denote column j of M. In describing the algorithm $f_{i,j}$ denotes the current value of the (i,j) entry of F, not the original one. Imagine that a square array F is initialized to contain L and ends up holding $\hat{L}^*$.

Algorithm 4 (The Implicit Cholesky Transform with Shift)

```
For $j = 1, 2, \ldots, n-1$ repeat
    (a) Overwrite the current $(j,j)$ entry $f_{j,j}$ with $d_j := \sqrt{f_{j,j}^2 - \tau^2}$
    (b) Acting on rows $j$ through $n$ find an orthogonal triangle preserving
    matrix $H_j$ such that $H_j f_{*j} = e_j \|f_{*j}\|$ and apply $H_j$ to those rows.
$f_{n,n} := d_n := \sqrt{f_{n,n}^2 - \tau^2}$.
```

Remark 1 An efficient way to apply H_j is to use fast Givens rotations. In this case F is held as two arrays Δ^2 and $\tilde{F}$ where $\tilde{F}$ is unit triangular and $L = \Delta\tilde{L}$.

Remark 2 The transformation alters the singular values and the left singular vectors but the right singular vectors of L are preserved as the left singular vectors of $\hat{L}$.

Remark 3 Note that the $2n \times 2n$ matrices are introduced to display the mathematical relationships. In practice the code resembles the program for Algorithm 2 except for the non-orthogonal modification of the diagonal entries. It is preferable to use triangle preserving rotations.

Remark 4 When shifts are used conclusion (b) in Theorem 1 fails but the other two results still hold.

Theorem 3 *Let $\hat{L}$ denote the implicit Cholesky transform of L with shift $\tau(< \sigma_{min}[L])$. Let d_i, $i = 1, \ldots, n$ be the intermediate diagonal entries described in Algorithm 4. Then*

$$\left(\sum_{k=1}^{n} d_k^{-2}\right)^{-1} - \tau^2 \le \sigma_{min}^2[\hat{L}] \le \min d_k^2$$

Proof: Part (a) in each step of Algorithm 4 cannot increase any singular value but may decrease some of them. If F denotes the square array that holds L initially and $\hat{L}^*$ finally then Part (a) is equivalent to reducing the (j,j) entry of FF^* by τ^2. At the end of Part (a) of Step j the array F has the form (shown with $n = 6$, $j = 4$)

$$F(4) = \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & d_k & & \\ & & & \times & \times & \\ & & & \times & \times & \times \end{pmatrix} \quad , \quad F(1) = L \quad , \quad F(n) = \hat{L}^*$$

Part (b) preserves singular values.

$$\sigma_n[\hat{L}] \le \sigma_n[F(j)] \le \|e_j^* F(j)\| = \|d_j e_j^*\| = d_j.$$

To derive the first inequality return to the $2n \times n$ arrays and exact orthogonal transforms.

Denote by Q_k the product of all the orthogonal transformations that deliver $F(k)$. Moreover let $(x_k^* \;\; y_k^*)$ denote row k of Q_k. Then $\|x_k\|^2 = 1 - \|y_k\|^2 \le 1$ and

$$x_k^* L = (x_k^* \;\; y_k^*) \begin{pmatrix} L \\ 0 \end{pmatrix} = e_k^* F(k) = d_k e_k^*.$$

Invert and transpose to find

$$d_k^{-1} x_k = L^{-*} u_k,$$

$$d_k^{-2} \|x_k\|^2 = [(L^* L)^{-1}]_{k,k}.$$

Thus

$$\sum_{k=1}^{n} d_k^{-2} \ge \sum_{k=1}^{n} d_k^{-2} \|x_k\|^2 = \text{trace}[(L^* L)^{-1}].$$

By Theorem 2, the Newton approximation from 0 to $\sigma_{min}^2[L]$ exceeds $(\sum_{k=1}^n d_k^{-2})^{-1}$. Subtract τ^2 to obtain a lower bound on $\sigma_{min}^2[\hat{L}]$. •

Repetition of Algorithm 4 will not deliver singular vectors because after two applications both right and left singular vectors have been lost. Algorithm 4 is an efficient way to find a few small singular values when reduction to bidiagonal form is not warranted. The next section shows one way to find singular vectors and use shifts.

9 Computation of Singular Vectors Using Shifts

Let R be upper triangular with positive diagonal. The goal is to compute some or all of R's singular values and the corresponding right singular vectors $v_1, v_2, \ldots, v_n$; $R^* R v_j =$

$v_j\sigma_j^2$. If any matrix M is multiplied on the right by an orthogonal matrix J then M's right singular vectors are transformed by J^* since

$$MJ = U\Sigma V^* J = U\Sigma(J^* V)^*.$$

Each step in Algorithm 5 makes two transformations; one of them modifies the vectors but makes no shift, the other preserves the (right) vectors but reduces the singular values.

Algorithm 5 (Implicit Shifted Cholesky for Singular Vectors of R)

```
Set $V = I$; $j := n$
While $j > 0$ do
    Repeat until the matrix is singular,
        (a) Transform $R$ into lower triangular form $L$ by plane rotations on the right.
        (b) Apply the transpose of each rotation to $V$.
        (c) Select a shift.
        (d) Transform $L$ into $\hat{R}$ by plane rotations on the left with shift $\tau$, as in Algorithm 4.
        (the right singular vectors are unchanged)
    $j := j - 1$
```

Comment If the initial matrix is lower triangular then begin the algorithm with a (d) transformation.

10 Choice of Shifts

Choosing a useful shift in LR and qd type algorithms is not an easy task. One would like the shift to be no larger than σ_{min} but too much caution begets sluggish convergence. If a shift is too large then one or more entries in the new matrix become pure imaginary and at the next step entries become complex. To deal with this situation appears to be more trouble than it is worth. So, at present, we only record a Cholesky transform when it is real. Aggressive shifts that result in late failure do yield excellent new shifts; see Sections 10.3 and 11.3.

10.1 Johnson Shift

A lower bound for smallest singular value of R is given by

$$\sigma_{min}[R] \geq \max[0, \min_{j=1,n}\{r_{j,j} - \frac{1}{2}(\sum_{i=1}^{j-1}|r_{i,j}| + \sum_{k=j+1}^{n}|r_{j,k}|)\}].$$

See [12]. In practice we may confine the search for $\min_j$ over a few j values surrounding m where $d_m = \min_k d_k$. This heuristic often gives a good result but it does not guarantee a lower bound for the smallest singular value.

10.2 Aggressive Shifts

We expect the shift strategy for these algorithms to evolve with experience. It is safe to always use the lower bound in Theorem 3 but it seems a shame to make no use of the increasingly accurate upper bound. Failures are expensive but not disastrous.

Our rather simple strategy is to let lo $= (\sum_{k=1} d_k^{-2})^{-1/2} - \tau^2$, hi $= \min_k d_k$ and set

$$\tau = \mathrm{lo} + \alpha * (\mathrm{hi} - \mathrm{lo})$$

after the change in hi is less than 5%. A shift that avoids the cost of the Newton shift (when the bandwidth is small, perhaps) is to save the previous value of hi as oldhi and set

$$\tau := 2 * \mathrm{hi} - \mathrm{oldhi}.$$

10.3 Response to Failure

Suppose that the shift $\tau > \sigma_{min}[L]$ and that the Implicit Cholesky algorithm fails at step k with $\hat{l}_{k,k}^2 < 0$. Let us write the relevant matrices in partitioned form

$$L^*L - \tau^2 I = \begin{bmatrix} A & F \\ F^* & M \end{bmatrix}$$

where A is $(k-1)\times(k-1)$ and positive definite. The failure at step k of implicit Cholesky shows that the Schur complement of A

$$W = M - F^* A^{-1} F$$

has $W_{1,1} = \hat{l}_{1,1}^2 < 0$. Consequently W has negative eigenvalues. In [23] Wilkinson showed that

$$\text{new } \tau := \sqrt{\tau^2 + \lambda_{min}[W]}$$

is a safe, and accurate, shift for L. When $k = n$ and W is 1×1 we recover Rutishauser's late failure result. For the analysis of this case see Section 11.3.

An alternative response to failure at step k (especially when k is small) is to use the fact that $\sqrt{\tau^2 + W_{1,1}} < \tau$ is a safe shift for the leading $k \times k$ submatrix of $L^*L - \tau^2 I$. An application of the implicit Cholesky transform with the new shift will fail, if at all, at some step after k. This shift is less work than the one Wilkinson suggested (especially when $k \approx n/2$) but is not guaranteed to give success.

The cautious response to early failure is to abandon τ and use a lower bound on σ_{min} as the next shift. At present we do not have comparisons of the effectiveness of these two alternatives. For late, or almost late failure (when W is of small order) Wilkinson's safe shift seems preferable.

11 Convergence

11.1 Linear Convergence

We have stressed the fact that all the algorithms that flip triangular matrices using one sided orthogonal transformations are mathematically equivalent to the unshifted LR Cholesky algorithm introduced by Rutishauser in [16]. Consequently his proofs of convergence suffice to cover all these variations and they are more elegant than the ones provided in these more recent studies. We cannot resist the temptation to show how simple the arguments can be. A little notation is needed. Let

$$R^* = \begin{pmatrix} S & 0 \\ H & E \end{pmatrix} \quad , \quad \hat{R} = \begin{pmatrix} \hat{S} & \hat{H} \\ 0 & \hat{E} \end{pmatrix}$$

be block upper triangular, S is $k \times k$, and let

$$R^* = Q\hat{R}$$

with $Q^* = Q^{-1}$. There is no need to partition Q. Since premultiplication by an orthogonal preserves inner products between columns we obtain three fundamental relations:

$$SS^* + HH^* = \hat{S}^*\hat{S} \tag{3}$$

$$EE^* = \hat{H}^*\hat{H} + \hat{E}^*\hat{E} \tag{4}$$

$$HE^* = \hat{S}^*\hat{H}. \tag{5}$$

Since HH^* and $\hat{H}^*\hat{H}$ are positive semi-definite it follows that

$$\sigma_i[S] \le \sigma_i[\hat{S}] \quad , \quad i = 1, \ldots, k \tag{6}$$

$$\sigma_i[E] \ge \sigma_i[\hat{E}] \quad , \quad i = 1, \ldots, n-k \tag{7}$$

and

$$\|\hat{H}\| \le \|E\| \, \|\hat{S}^{-1}\| \, \|H\|. \tag{8}$$

From (6)

$$\|\hat{S}^{-1}\| = \sigma_k[\hat{S}]^{-1} \le \sigma_k[S]^{-1} = \|S^{-1}\| \tag{9}$$

and thus

$$\|\hat{H}\| \le \|E\| \, \|S^{-1}\| \, \|H\|. \tag{10}$$

These results hold for any partition of R, i.e. any admissible choice of k. Provided that R is undecomposable, e.g. no H block vanishes, then as the algorithm proceeds the $(1,1)$ blocks must increase and $(2,2)$ blocks must decrease when measured by the Frobenius norm. It follows that the large singular values must migrate to the $(1,1)$ block and the small ones must descend to the $(2,2)$ block. So if all the singular values are distinct then the matrix must converge to $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$. However in the presence of multiple singular values, and finite precision arithmetic too, the algorithm need not converge.

Equations (8) and (10) allow us to estimate the rate of convergence. If the partition size k is chosen so that there is a gap, $\sigma_k >> \sigma_{k+1}$, then at some point in the process

$$\|E\|\,\|S^{-1}\| \approx \sigma_{k+1}/\sigma_k < 1 \quad \text{(although} \quad \|E\|\,\|S^{-1}\| \geq \sigma_{k+1}/\sigma_k)$$

and then the $(1,2)$ blocks diminish monotonically in norm. Nevertheless convergence of $\|H\|$ is linear.

Mathias and Stewart give a sufficient condition for monotonicity to set in. In our notation their condition is that

$$\|H\| + \|E\| < \sigma_k[R]. \tag{11}$$

To see why this suffices consider the matrix

$$M := \begin{pmatrix} 0 & S & 0 & 0 \\ S^* & 0 & H & 0 \\ 0 & H^* & 0 & E \\ 0 & 0 & E^* & 0 \end{pmatrix}$$

whose eigenvalues are $\pm\sigma_i[R], i = 1, \ldots, n$.

By the Wielandt-Hoffmann theorem applied to M the singular values of R may be paired with those of S and E in such a way that the difference of any pair is at most $\|H\|$. By (11)

$$\sigma_k[R] - \sigma_i[E] \geq \sigma_k[R] - \|E\| > \|H\| \tag{12}$$

and so the largest k singular values of R cannot be paired with any from E. Thus for some $j \leq k$

$$\sigma_k[S] \geq \sigma_j[R] - \|H\| > \sigma_k[R] - \|H\| \tag{13}$$

and by (11) and (13),

$$\|E\|\,\|S^{-1}\| \leq \frac{\|E\|}{\sigma_k[R] - \|H\|} \tag{14}$$

When this stage is reached, and (11) holds, then E's singular values approximate the small singular values of R to $O(\|H\|^2)$.

To see this consider the symmetric indefinite matrix M given above and consider the subspace

$$\text{Range} \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ I & 0 \\ 0 & I \end{pmatrix}.$$

The projection of M onto this subspace is represented by

$$\begin{pmatrix} 0 & E \\ E^* & 0 \end{pmatrix}$$

and the matrix residual F is given by

$$F = M \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ I & 0 \\ 0 & I \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ I & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} 0 & E \\ E^* & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ H & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

Thus $\|F\| = \|H\|$. If γ is the gap between E's singular values and the largest k singular values of R, i.e. $\gamma = \sigma_k[R] - \|E\|$ then, by Chap. 11 of [15],

$$|\sigma_i[E] - \sigma_{k+i}[R]| \leq \|F\|^2/\gamma = \|H\|^2/\gamma \quad , \quad i = 1, \ldots, n-k.$$

11.2 Quadratic Convergence

Sections 7 and 8 show that the Newton approximation τ^2 to $\sigma_{min}^2(L)$ from 0 is easily computed at the end of the LR transform. By the monotonicity of Newton's iteration from outside the zero set we know that τ is a safe shift. By Theorem 3, the same formula $(\sum d_k^{-2})^{-\frac{1}{2}}$, when used at the end of a shifted Cholesky step, still provides a lower bound on $\sigma_{min}[\hat{L}]$ but a poorer one than Newton's. However, shifts are non-restoring in the implicit Cholesky process and consequently the shifts will tend to 0 and the shift formula will degrade only a little. Nevertheless we have not proved quadratic convergence for this shift formula in Algorithm 4.

On the other hand Algorithm 5 does use Newton's shift and convergence is quadratic but each step costs twice as much as a step of Algorithm 4. However the goals of these two algorithms are not the same.

11.3 Higher Order Convergence

Rutishauser observed that when the matrix has nearly revealed the smallest singular value in the (n, n) position then there is a shift strategy that generates cubic convergence. The key requirement is what he calls *late breakdown*. Suppose that $\tau = l_{n,n}$ exceeds $\sigma_{min}[L]$ by so little that a negative pivot d_n^2 occurs only at the last step. It turns out that $\sqrt{l_{n,n}^2 + d_n^2}$ is a safe and accurate shift for L. So the strategy consists of making a copy of L, transforming with shift $l_{n,n}$ and then discarding intermediate quantities except for d_n^2. Then transform L with shift $\sqrt{l_{n,n}^2 + d_n^2}(< l_{n,n})$.

Neither Rutishauser nor Wilkinson [23] emphasize that to employ this strategy it is necessary to save L and discard all the intermediate quantities arising from using the shift $l_{n,n}$. Thus the cost of finding the good shift is one whole iteration. Perhaps that is why this shift is more talked about than used. In fact there is a small interval of shifts $\tau(\leq l_{n,n})$ which produce a rate of convergence at least as good as from $l_{n,n}$.

We present here the simple derivation in the case where the initial shift τ is taken as $l_{n,n}$, because the proof of the more general result is messier.

Write L in partitioned form as

$$L = \begin{bmatrix} \tilde{L} & 0 \\ b^* & l_{n,n} \end{bmatrix}.$$

The assumption is that $\sigma_{min} = \sigma_{nin}[L] < \sigma_{min}[\tilde{L}]$. Consequently the last pivot in the triangular factorization of $L^*L - \sigma_{min}^2 I$ must vanish. This gives

$$\text{Fact 1.}\quad l_{n,n}^2 - \sigma_{min}^2 - l_{n,n}^2 b^*[V - \sigma_{min}^2 I_{n-1}]^{-1} b = 0$$

where $V := \tilde{L}^*\tilde{L} + bb^*$.

When the shift $\tau = l_{n,n}$ is used the late failure assumption says that the last pivot is the only negative one. This gives

$$\text{Fact 2.}\quad d_n^2 := 0 - l_{n,n}^2 b^*[V - l_{n,n}^2 I_{n-1}]^{-1} b < 0.$$

Hence

$$l_{n,n}^2 + d_n^2 = \sigma_{min}^2 + l_{n,n}^2\{b^*[V - \sigma_{min}^2 I_{n-1}]^{-1} b - b^*[V - l_{n,n}^2 I_{n-1}]^{-1} b\}$$

By Hilbert's first resolvent identity (see p 90 of [3])

$$l_{n,n}^2 + d_n^2 = \sigma_{min}^2 + l_{n,n}^2(\sigma_{min}^2 - l_{n,n}^2)\{b^*[V - \sigma_{min}^2 I_{n-1}]^{-1}[V - l_{n,n}^2 I_{n-1}]^{-1} b\}$$

Now use Fact 1 again to find that

$$l_{n,n}^2 + d_n^2 = \sigma_{min}^2 - l_{n,n}^4 b^*[V - \sigma_{min}^2 I_{n-1}]^{-1} b\{b^*[V - \sigma_{min}^2 I_{n-1}]^{-1}[V - l_{n,n}^2 I_{n-1}]^{-1} b\}.$$

It is more informative to write $b = \|b\|\tilde{b}$ so that

$$l_{n,n} + d_n^2 = \sigma_{min}^2 - (l_{n,n}\|b\|)^4 \tilde{b}^*[V - \sigma_{min}^2 I_{n-1}]^{-1}\tilde{b}\{\tilde{b}[V - \sigma_{min}^2 I_{n-1}]^{-1}[V - l_{n,n}^2 I_{n-1}]^{-1}\tilde{b}\}.$$

The quantities involving $\tilde{b}$ are bounded by $1/(\sigma_{n-1}^2[L] - \sigma_n^2[L])^3$ as $l_{n,n}\|b\| \to 0$.

Although this result is interesting we doubt that the $l_{n,n}$ trial shift is cost effective in the full triangular case. Wilkinson and Rutishauser did not know the bounds on $\sigma_n[L]$ given by Theorem 3, and so we can use more refined strategies than they did.

12 A Preconditioner for Jacobi

Jacobi invented his famous eigenvalue algorithm (given in [11]) as a preconditioner for what we now call the point Jacobi method for solution of linear (symmetric) equations. See [9], [10]. One might ask the wisdom of designing a preconditioner for another "preconditioner". The answer is that the classical Jacobi method in which the largest offdiagonal element is chosen as the doomed element is no longer the preferred version for computation of eigenvalues. Instead the row (or column) cyclic annihilations are used in Jacobi algorithms and this strategy does not have good initial convergence properties, see Fernando [6], although asymptotically the convergence is quadratic. Hence the use of a preconditioner for cyclic Jacobi is not a futile effort.

The flipping algorithm (Algorithm 2) has a non-trivial application in one-sided Jacobi algorithms belonging to the Hestenes family, see [8]. Veselić and Hari [21] have shown that one step of the **Cholesky** algorithm with diagonal pivoting helps the convergence of Hestenes algorithm for some cases. That is, the **Cholesky** algorithm can be used to precondition the Hestenes type SVD algorithms. By recognizing that the LR algorithm can be implemented using orthogonal or unitary transformations, it is possible to execute extra preconditioning steps. Jacobi algorithms are more appropriate at the later stages as they possess quadratic convergence whereas LR algorithms without shifts are linearly convergent, so only a few preconditioning steps should be used. It is not yet known whether Jacobi will beat implicit Cholesky with shifts. This preconditioner can be also used for the Jacobi method due to Kogbetliantz for computation of the SVD. If the matrix is triangular, the Kogbetliantz method retains that structure while Hestenes does not. See Fernando [6].

Let us indicate how the preconditioning step may be applied more than once. Let A be symmetric positive definite. The authors mentioned above perform a Cholesky factorization with diagonal pivoting to produce L where

$$P^*AP = LL^*.$$

The one-sided Jacobi algorithm applies a sequence of plane rotations, on the right, to an array F which is initialized to either L or L^*. Each rotation makes the associated columns of F orthogonal. On exit the norms of F's columns are the singular values of L and the columns themselves are eigenvectors of A. Veselić and Hari observed that initializing F to L, instead of L^*, is equivalent to performing Jacobi on L^*L rather than $LL^* = A$. In other words the Jacobi process is preconditioned (implicitly) by one LR transform. This preconditioning sometimes improves convergence dramatically. Why not do another step of preconditioning ? Veselić and Hari were unwilling to discard the accuracy inherent in Jacobi by explicitly forming L^*L in order to compute the Cholesky factor $\hat{L}$ ($L^*L = \hat{L}\hat{L}^*$) but as explained in this paper, there is no need for explicit multiplication. Algorithm 2 yields $\hat{L}$, without forming L^*L, using plane rotations exclusively. Now F may be initialized to $\hat{L}$ instead of L. The process may be repeated and in fact, should be.

Let us mention a subtle point concerning preconditioning for a right sided Jacobi process applied to a matrix F. Let $A = LL^*$ and let $L = U\Sigma V^*$. If F is initialized to L^* then, on exit, the columns of F when normalized give V. If F is initialized to L then, on exit, the columns of F yield U. If we do an extra preconditioning step from L to $\hat{L}$ then initializing F to $\hat{L}$ will eventually yield V. Another preconditioning step $\hat{L} \rightarrow \tilde{L}$ and initializing F to $\tilde{L}$ will eventually yield U and again there is no need to accumulate rotations and this feature saves both storage and arithmetic operations. However, the accuracy and especially the orthogonality of the vectors computed this way might be poorer than for the accumulated vectors. Nevertheless, the moral of the story is that it pays to use an even number of preconditioning steps for one vector set, an odd number for the other.

For those who are appalled at the thought of preceding a Jacobi process by a non-orthogonal transformation $A \rightarrow LL^*$, there is a remedy. Compute the QR factorization of Hermitian A, ignoring A's symmetry but using column pivoting or some other rank revealing permutation. Then initialize the matrix F to R or R^*. In exact arithmetic this preconditioning is equivalent to using the Cholesky factor of A^2. The advantages of this more expensive preconditioner is that there is no need for A to be positive definite. Furthermore, the QR step can be done in parallel machines, including systolic arrays, while the Cholesky step cannot be executed so easily in parallel. If the matrix is not sign definite, the signs of A's eigenvalues can be recovered on exit by using the eigenvectors formed in the Jacobi process. Convergence will be much faster than when the preconditioner is a Cholesky factor because the convergence factors are squared as our numerical examples reveal.

In the above algorithm, we assumed that the matrix is Hermitian. If the matrix is skew-Hermitian, the same algorithm can be used by noting that now the eigenvalues are conjugate imaginary pairs whose magnitudes are given by the singular values. So there is no concern about recovering the signs of the eigenvalues.

Once we have taken the fateful step of contemplating a preconditioner for a Jacobi process we are lead inexorably to the message of this paper. Why not use the implicit Cholesky algorithm with shifts as a preconditioner? There is no loss of accuracy. The next question is: if the shifts are well chosen why switch to Jacobi? Time will tell.

13 Numerical Examples

The goals of this study are (i) to point out the feasibility and the desirability of shifting in the implicit Cholesky algorithm, (ii) to present new computable upper and lower bounds on singular values. We are not presenting a particular implementation and so content ourselves with offering some simple illustrations.

Two types of triangular matrix were used as input. Any symmetric positive definite matrix A may be decomposed as $A = LL^*$ and as $A = QR$. Thus R is the Cholesky factor of A^2 and its singular values are the squares of those of L^*. We present results on triangular matrices arising from a 10×10 reverse Hilbert matrix and from a 20×20 Toeplitz matrix ($A(i,j) = n - |i-j|$, $i = 1,\ldots,n$, $j = 1,\ldots,n$).

For each triangular matrix we tried to present the results of 3 shifting strategies: no shift, Newton shift, and a simple aggressive shift described in Section 10.2.

Before commenting on the results we discuss our stopping criterion. Let

$$R = \begin{pmatrix} S & H \\ O & E \end{pmatrix}$$

be the current matrix. We declare H negligible when

$$||H|| < tol * macheps * ||\text{original } R||.$$

For the examples given here E was 1×1 and $||\cdot|| = ||\cdot||_\infty$. Sometimes convergence would occur sooner if we allowed E to have larger dimension but when E is 1×1 we do learn the number of steps needed to find the singular values in order.

We began with $tol = 1$ (i.e. maximal absolute accuracy) and used a Sun Sparc Workstation ELC with $macheps \doteq 2 \times 10^{-16}$. With no shift the algorithm failed to converge after 2 minutes on the Toeplitz example. It still failed with $tol = 10^6$ so we ended up using $tol = 10^8$ just to get a comparison, see Tables 1 and 2. Even with this value of tol the Cholesky factor of the Toeplitz matrix required 1179 steps with no shift as compared with 110 for the Newton shift and 84 for the aggressive shift.

The Toeplitz example, see Tables 1, 2, 3, and 4, shows clearly the potential benefits of shifting. The smaller singular values, see Tables 9 and 10, are not very small and their ratios σ_{i+1}/σ_i are not very small either. On the other hand the σ_i differ in the second decimal so that they could not be considered clustered. As is clear from Tables 1 and 2 the $(19, 20)$ entry is the last to become negligible. In fact, with no shift, the singular values converge in the order $\sigma_1, \sigma_2, \ldots, \sigma_{20}$. However our purpose is to show how shifts can help and 465 steps are needed to make the $(19, 20)$ entry go below $10^8 * macheps * norm$. In contrast the Cholesky factor of the reverse Hilbert segment has much smaller values for σ_{i+1}/σ_i, see Tables 13 and 14, and the improvements from shifting are much less dramatic, see Tables 7 and 8. Tables 3, 4, 5, and 6 show two things: the potential power of good shifts and the slow down caused by the poorer ratios associated with the Cholesky triangle, see Tables 9, 10, 11, and 12.

The quality of our bounds on σ_{min} is illustrated in Figs. 1 and 3. Recall that in Algorithm 5 a shift is only used on alternate flips when singular vectors are requested. The lower bound in the shifted flip is poorer than in the unshifted one and that accounts for the step like appearance of the lower bound in Fig. 1.

Note that the upper bound is a much better approximation than the lower one. Finally, Figs. 2 and 4, we give a snapshot of the way in which $||H||_\infty$ declines under two shift strategies.

Even these preliminary results suggest that implicit Cholesky with *good shifts* is not just a tool for low accuracy applications but can be used in high accuracy, LAPACK style, mode as well. It may be the method of choice whenever reduction to bidiagonal form is too much trouble.

References

[1] L. P. Ammann. Robust singular value decompositions: A new approach to projection pursuit. Technical report, Erik Jonsson School of Engineering and Computer Science, The University of Texas at Dallas, 1992.

[2] S. Chandrasekaran and I.C.F. Ipsen. Analysis of a QR algorithm for computing singular values. Research Report 917, Department of Computer Science, Yale University, 1992.

[3] Francoise Chatelin. *Spectral Approximation of Linear Operators.* Academic Press, New York, 1983.

[4] E. M. Dowling, L. P. Ammann, and R. D. DeGroat. A TQR-iteration based adaptive SVD for real time angle and frequency tracking. Technical report, Erik Jonsson School of Engineering and Computer Science, The University of Texas at Dallas, 1992.

[5] D. K. Faddeev, V. N. Kublanovskaya, and V. N. Faddeeva. Sur les systèmes linèaires algèbraiques de matrices rectangulaires et mal-conditionnèes. *Colloq. Int. du C.N.R.S. Besançon 1966*, No. 165:161–70, 1968.

[6] K. Vince Fernando. Linear convergence of row cyclic Jacobi and Kogbetliantz methods. *Numer. Math.*, 56:73–91, 1989.

[7] K. Vince Fernando and Beresford N. Parlett. Accurate singular values and differential qd algorithms. Technical Report PAM-554, Centre for Pure and Applied Mathematics, University of California at Berkeley, 1992.

[8] M. R. Hestenes. Inversion of matrices by biorthogonalization and related results. *J. SIAM*, 6:51–90, 1958.

[9] C. G. J. Jacobi. On a new way of solving the linear equations that arise in the method of least squares (Translated by G. W. Stewart). Technical Report UMIACS-TR-92-42, CS-TR 2887, Dept. of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, 1992.

[10] C. J. G. Jacobi. Ueber eine neue Auflösungsart der bei der Methode der kleinsten Quadrate vorkommenden lineären Gleichungen. *Astronomische Nachrichten*, 22:297–306, 1845.

[11] C. J. G. Jacobi. Über ein leichtes Verfahren die in der Theorie der Säcularstörungen vorkommenden Gleichungen numerisch aufzulösen. *Journal reine angew Mathematik (Crelle's Journal)*, 30:51–94, 1846.

[12] Charles. R. Johnson. A Gersgorin-type lower bound for the smallest singular value. *Linear Algebra and Its Applications*, 112:1–7, 1989.

[13] R. Mathias and G. W. Stewart. A block QR algorithm and the singular value decomposition. Technical Report UMIACS-TR-91-38, CS-TR 2626, Dept. of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, 1991.

[14] M. Moonen, P. Van Dooren, and F. Vanpoucke. On the QR algorithm and updating the SVD and URV decompositions in parallel. Technical report, ESAT, Katholieke Universiteit Leuven, Belgium, 1992.

[15] Beresford. N. Parlett. *The Symmetric Eigenvalue Problem.* Prentice-Hall, Englewood Cliffs, NJ, 1980.

[16] H. Rutishauser. Solution of eigenvalue problems with the LR-transformation. *Nat. Bur. Standards Appl. Math. Series*, 49:47–81, 1958.

[17] H. Rutishauser. *Lectures on Numerical Mathematics.* Birkhäuser, Boston, 1990.

[18] H. Rutishauser and H. R. Schwarz. The LR transformation method for symmetric matrices. *Numer. Math.*, 5:273–289, 1963.

[19] H. R. Schwarz, H. Rutishauser, and E. Stiefel. *Numerical Analysis of Symmetric Matrices.* Prentice-Hall, Englewood Cliffs, NJ, 1973.

[20] G. W. Stewart. Updating a rank-revealing ulv decomposition. *SIAM Journal on Matrix Analysis and Applications*, 14:494–499, 1993.

[21] K Veselić and V. Hari. A note on a one-sided Jacobi algorithm. *Numer. Math.*, 56:627–633, 1989.

[22] U. von Matt. *Large constrained quadratic problems.* Ph.d. thesis, Institute for Scientific computing, ETH, Zurich, Switzerland, 1993.

[23] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Clarendon Press, Oxford, 1965.

Toeplitz Triangle: QR Mode: Vectors										
i	20	19	18	17	16	15	14	13	12	11
No Shift	455	456	457	458	459	460	461	462	463	464
Newton	35	36	45	52	58	60	61	66	69	70
Aggressive	17	18	25	31	37	38	39	45	49	54

Table 1: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 10^8$.**

Toeplitz Triangle: QR Mode: Vectors (cont.)										
i	10	9	8	7	6	5	4	3	2	1
No Shift	465	466	467	468	469	470	471	472	473	473
Newton	71	72	73	74	75	76	77	78	79	79
Aggressive	57	58	59	60	61	62	63	64	65	65

Table 2: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 10^8$.**

Toeplitz Triangle: QR Mode: Vectors										
i	20	19	18	17	16	15	14	13	12	11
Newton	39	46	59	72	84	86	97	108	118	124
Aggressive	19	26	35	44	52	54	61	68	74	79

Table 3: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 1$.**

Toeplitz Triangle: QR Mode: Vectors (cont.)										
i	10	9	8	7	6	5	4	3	2	1
Newton	130	131	132	133	134	135	136	137	138	138
Aggressive	84	88	93	94	97	98	99	100	101	101

Table 4: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 1$.**

Toeplitz Triangle: Cholesky Mode: Vectors										
i	20	19	18	17	16	15	14	13	12	11
Newton	54	62	76	88	101	102	116	127	138	148
Aggressive	20	29	37	44	53	54	62	69	76	82

Table 5: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 1$.**

Toeplitz Triangle: Cholesky Mode: Vectors (cont.)										
i	10	9	8	7	6	5	4	3	2	1
Newton	158	164	171	172	173	174	175	176	177	177
Aggressive	89	95	100	102	108	112	115	116	117	117

Table 6: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 1$.**

Reverse Hilbert Triangle: QR Mode: Vectors										
i	10	9	8	7	6	5	4	3	2	1
No Shift	3	5	8	11	13	16	19	23	31	31
Newton	3	5	8	11	13	16	19	23	29	29
Aggressive	3	5	8	11	13	16	19	23	27	27

Table 7: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 1$.**

Reverse Hilbert Triangle: Cholesky Mode: Vectors										
i	10	9	8	7	6	5	4	3	2	1
No Shift	10	14	17	21	25	29	33	40	53	53
Newton	5	12	17	21	25	29	33	39	46	46
Aggressive	5	11	16	20	24	28	32	36	41	41

Table 8: **Steps needed to find $\{\sigma_n, \ldots, \sigma_i\}$, $tol = 1$.**

Toeplitz Triangle: QR										
i	20	19	18	17	16	15	14	13	12	11
Values	0.503	0.512	0.529	0.552	0.586	0.629	0.688	0.762	0.865	0.995
Ratio	0.982	0.969	0.957	0.943	0.931	0.915	0.903	0.881	0.869	0.839

Table 9: **Singular Values**

Toeplitz Triangle: QR (cont.)										
i	10	9	8	7	6	5	4	3	2	1
Values	1.19	1.43	1.83	2.38	3.41	5.00	9.17	17.2	81.2	270.
Ratio	0.827	0.783	0.769	0.697	0.682	0.545	0.532	0.212	0.300	

Table 10: **Singular Values**

Toeplitz Triangle: Cholesky										
i	20	19	18	17	16	15	14	13	12	11
Values	0.709	0.716	0.727	0.743	0.765	0.793	0.829	0.873	0.930	0.998
Ratio	0.991	0.984	0.978	0.971	0.965	0.956	0.950	0.939	0.932	0.916

Table 11: **Singular Values**

Toeplitz Triangle: Cholesky (cont.)										
i	10	9	8	7	6	5	4	3	2	1
Values	1.09	1.20	1.35	1.54	1.85	2.24	3.03	4.15	9.01	16.4
Ratio	0.909	0.885	0.877	0.835	0.826	0.739	0.730	0.461	0.548	

Table 12: **Singular Values**

Reverse Hilbert Triangle: QR										
i	10	9	8	7	6	5	4	3	2	1
Values	.11e-12	.23e-10	.21e-8	.12e-6	.47e-5	.13e-3	.25e-2	.36e-1	.34	1.8
Ratio	.48e-02	.11e-01	.17e-1	.26e-1	.37e-1	.51e-1	.71e-1	.10	.20	

Table 13: **Singular Values**

Reverse Hilbert Triangle: Cholesky										
i	10	9	8	7	6	5	4	3	2	1
Values	.33e-6	.48e-05	.46e-4	.35e-3	.22e-2	.11e-1	.50e-1	.19	.59	1.3
Ratio	.69e-1	.10	.13	.16	.19	.23	.27	.32	.44	

Table 14: **Singular Values**

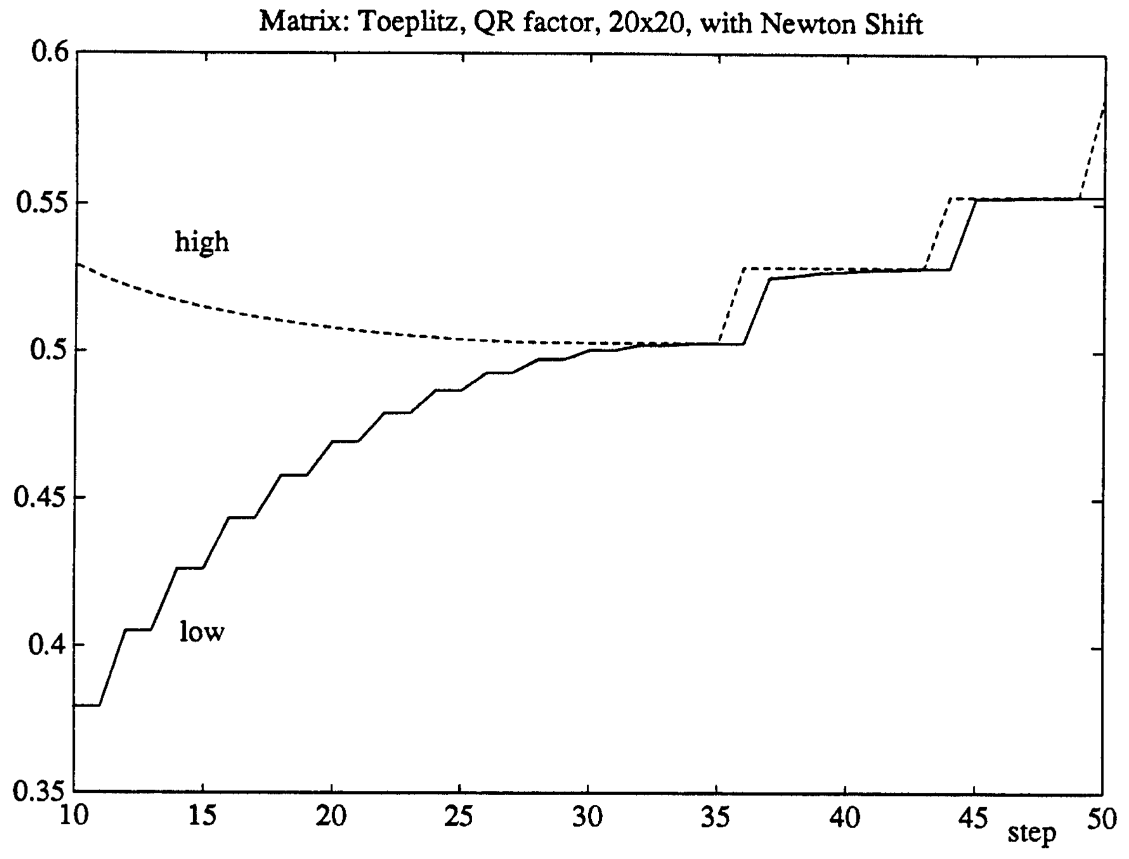

Fig.1 Bounds for Singular Values, tol = 1e8

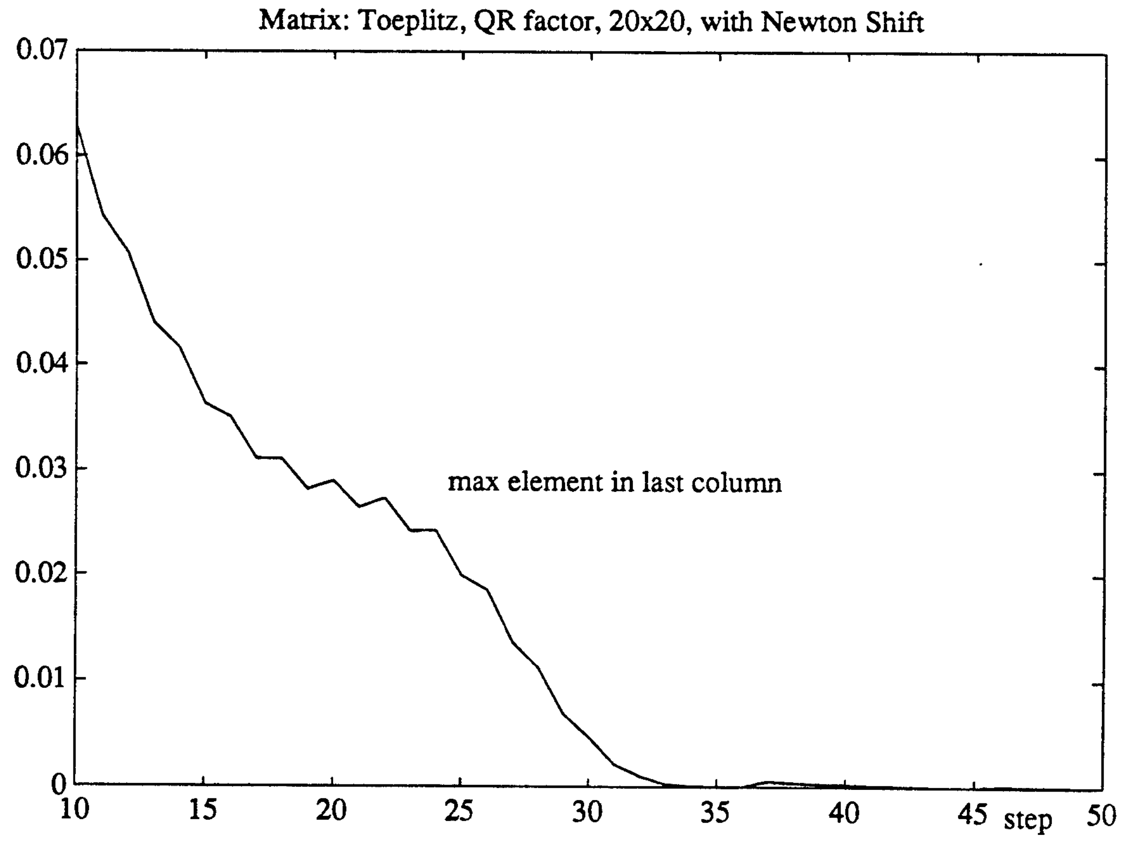

Fig. 2 Decline of max element in last column, tol = 1e8

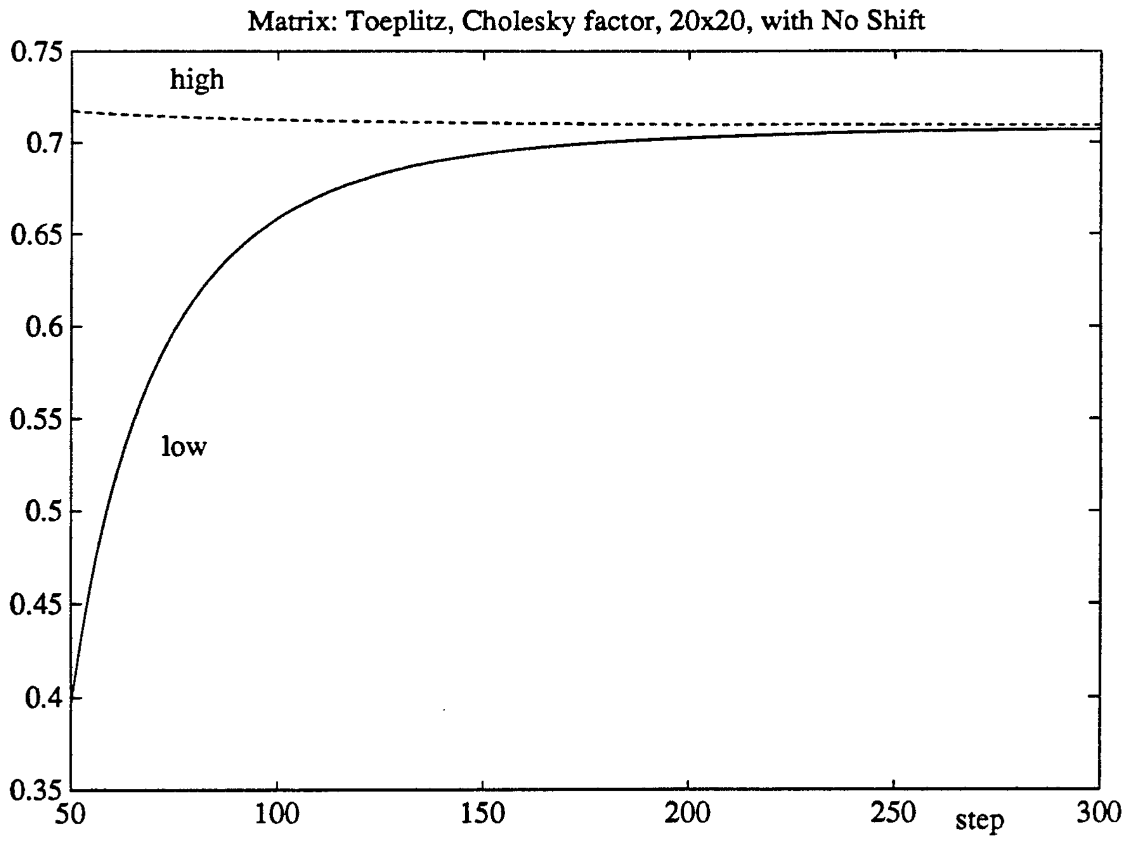

Fig. 3 Bounds for Singular Values, tol = 1e8

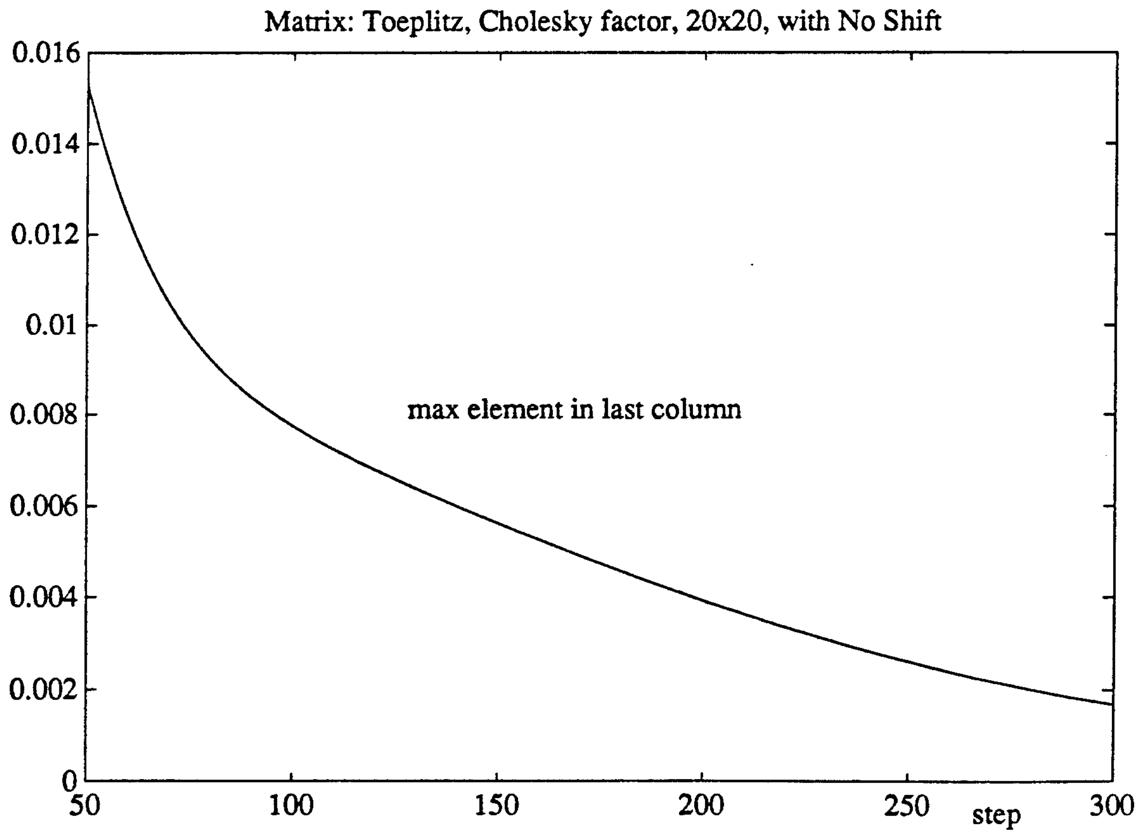

Fig. 4 Decline of max element in last column, tol = 1e8